SQL #2

Wongyu Kim

5th Week

Agenda

- Subqueries
- Grouping
- HAVING Clause
- Outer Joins

Subqueries

Use output of other queries.

e.g.) Find the names of departments that there exist employees.

Employees

EID	Name	DepartmentID
1	Alice	101
2	Bob	102
3	Charlie	103

Departments

DID	Name
101	HR
102	Sales
103	IT
104	Marketing

Subqueries

Use output of other queries.

e.g.) Find the names of departments that there exist employees.

Employees

EID	Name	DepartmentID
1	Alice	101
2	Bob	102
3	Charlie	103

Departments

DID	Name
101	HR
102	Sales
103	IT
104	Marketing

```
SELECT Name
FROM Departments D
WHERE EXISTS (
        SELECT EID
        FROM Employees E
        WHERE E.DepartmentID = D.DID
)
```

```
SELECT Name
FROM Departments
WHERE DID IN (
        SELECT DepartmentID
        FROM Employees
)
```

Name
HR
Sales
IT

Grouping

- Compute aggregated values for each certain attribute.
- Template

```
SELECT exp-1, exp-2, ..., exp-n, aggregation_function(aggr. exp)
FROM tables
WHERE conditions
GROUP BY exp-1, exp-2, ..., exp-n
```

attributes

COUNT, SUM, AVG, MIN, MAX, …

All columns specified in the SELECT clause must appear in the GROUP BY clause, and vice versa.

Grouping – Example#1

Find total number of employees.

Employee

ID	Name	Department	Salary
1	Alice	HR	60000
2	Bob	Finance	70000
3	Charlie	Finance	80000
4	David	IT	90000
5	Eve	IT	85000
6	Frank	HR	62000

Grouping – Example#1

Find total number of employees.

Employee

ID	Name	Department	Salary
1	Alice	HR	60000
2	Bob	Finance	70000
3	Charlie	Finance	80000
4	David	IT	90000
5	Eve	IT	85000
6	Frank	HR	62000

```
SELECT COUNT(*)
FROM Employee;
```

COUNT(*)
6

Grouping – Example#2

Find the number of employees for each department.

Employee

ID	Name	Department	Salary
1	Alice	HR	60000
2	Bob	Finance	70000
3	Charlie	Finance	80000
4	David	IT	90000
5	Eve	IT	85000
6	Frank	HR	62000

Grouping – Example#2

Find the number of employees for each department.

Employee

ID	Name	Department	Salary
1	Alice	HR	60000
2	Bob	Finance	70000
3	Charlie	Finance	80000
4	David	IT	90000
5	Eve	IT	85000
6	Frank	HR	62000

```
SELECT Department, COUNT(*)
FROM Employee
GROUP BY Department;
```

Deparment	COUNT(*)
Finance	2
HR	2
IT	2

Grouping – Example#3

Find total, average, minimum, and maximum salaries of employees.

Employee

ID	Name	Department	Salary
1	Alice	HR	60000
2	Bob	Finance	70000
3	Charlie	Finance	80000
4	David	IT	90000
5	Eve	IT	85000
6	Frank	HR	62000

Grouping – Example#3

Find total, average, minimum, and maximum salaries of employees.

Employee

ID	Name	Department	Salary
1	Alice	HR	60000
2	Bob	Finance	70000
3	Charlie	Finance	80000
4	David	IT	90000
5	Eve	IT	85000
6	Frank	HR	62000

```
SELECT SUM(Salary) AS Total,
       AVG(Salary) AS Average,
       MIN(Salary) AS Minimum,
       MAX(Salary) AS Maximum
FROM Employee;
```

Total	Average	Minimum	Maxmimum
447000	74500	60000	9000

Having Clause

Filter to aggregated values

(e.g.) Find departments and their average salary that average salary is over 80000

Employee

ID	Name	Department	Salary
1	Alice	HR	60000
2	Bob	Finance	70000
3	Charlie	Finance	80000
4	David	IT	90000
5	Eve	IT	85000
6	Frank	HR	62000

Having Clause

Filter to aggregated values

(e.g.) Find departments and their average salary that average salary is over 80000

Employee

ID	Name	Department	Salary
1	Alice	HR	60000
2	Bob	Finance	70000
3	Charlie	Finance	80000
4	David	IT	90000
5	Eve	IT	85000
6	Frank	HR	62000

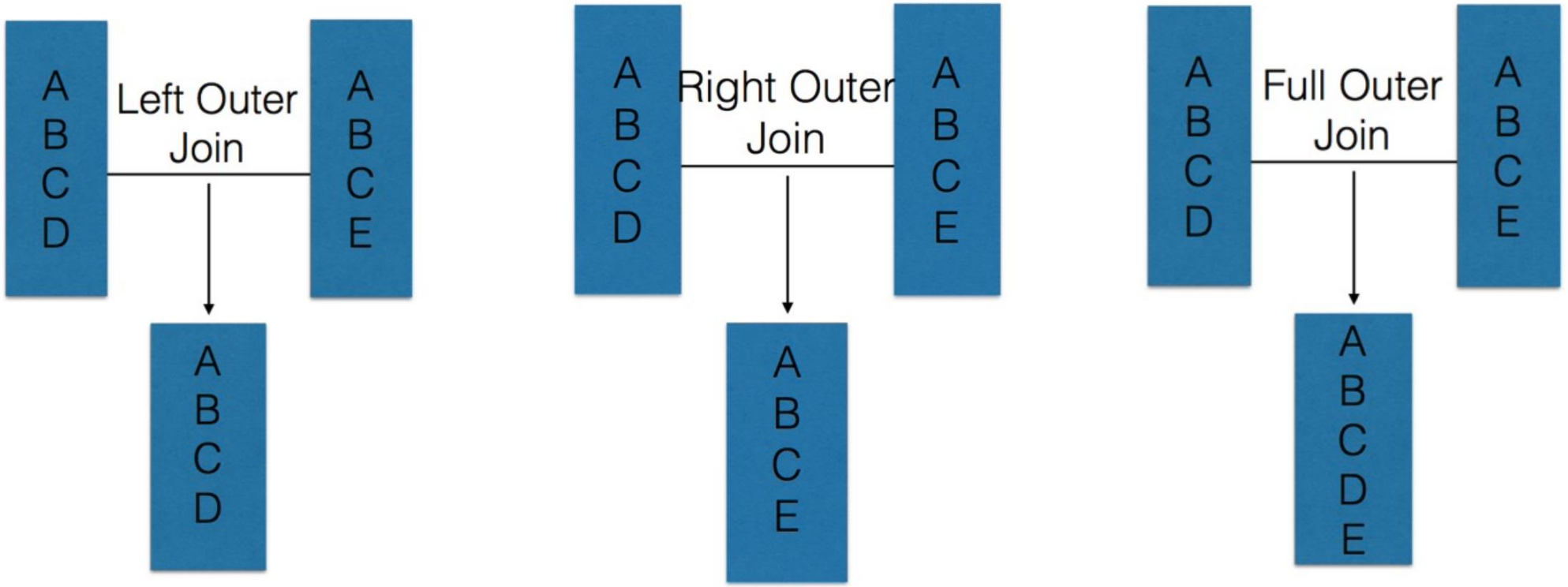
```
SELECT Department, AVG(Salary) AS Average
FROM Employee
GROUP BY Department
HAVING AVG(Salary) > 80000;
```

Department	Average
IT	87500

HR	*61000*
Finance	*75000*

Outer Join

- An extension of the join operation that avoids loss of information.
- Outer join preserves dangling tuples by padding them with NULL.



Outer Join - Example

Left Outer Join

TA

ID	Name
1	Pirate
2	Monkey
3	Ninja
4	Spaghetti

TB

ID	Name
1	Rutabaga
2	Pirate
3	Darth Vader
4	Ninja

```
SELECT TA.ID, TA.Name, TB.Name, TB.ID
FROM TA
LEFT JOIN TB
ON TA.Name = TB.Name;
```

Outer Join - Example

Left Outer Join

TA

ID	Name
1	Pirate
2	Monkey
3	Ninja
4	Spaghetti

TB

ID	Name
1	Rutabaga
2	Pirate
3	Darth Vader
4	Ninja

```
SELECT TA.ID, TA.Name, TB.Name, TB.ID
FROM TA
LEFT JOIN TB
ON TA.Name = TB.Name;
```

TA.ID	TA.Name	TB.Name	TB.ID
1	Pirate	Pirate	2
2	Monkey	NULL	NULL
3	Ninja	Ninja	4
4	Spaghetti	NULL	NULL

All values of Left Table(TA) are printed.

Dangling tuples

Outer Join - Example

Right Outer Join

TA

ID	**Name**
1	Pirate
2	Monkey
3	Ninja
4	Spaghetti

TB

ID	**Name**
1	Rutabaga
2	Pirate
3	Darth Vader
4	Ninja

```
SELECT TA.ID, TA.Name, TB.Name, TB.ID
FROM TA
RIGHT JOIN TB
ON TA.Name = TB.Name;
```

Outer Join - Example

Right Outer Join

TA

ID	Name
1	Pirate
2	Monkey
3	Ninja
4	Spaghetti

TB

ID	Name
1	Rutabaga
2	Pirate
3	Darth Vader
4	Ninja

```
SELECT TA.ID, TA.Name, TB.Name, TB.ID
FROM TA
RIGHT JOIN TB
ON TA.Name = TB.Name;
```

TA.ID	TA.Name	TB.Name	TB.ID
NULL	NULL	Rutabaga	1
1	Pirate	Pirate	2
NULL	NULL	Darth Vader	3
3	Ninja	Ninja	4

Dangling tuples

All values of Right Table(TB) are printed.

Outer Join - Example

Full Outer Join

TA

ID	Name
1	Pirate
2	Monkey
3	Ninja
4	Spaghetti

TB

ID	Name
1	Rutabaga
2	Pirate
3	Darth Vader
4	Ninja

```
SELECT TA.ID, TA.Name, TB.Name, TB.ID
FROM TA
FULL JOIN TB
ON TA.Name = TB.Name;
```

Outer Join - Example

Full Outer Join

TA

ID	Name
1	Pirate
2	Monkey
3	Ninja
4	Spaghetti

TB

ID	Name
1	Rutabaga
2	Pirate
3	Darth Vader
4	Ninja

```
SELECT TA.ID, TA.Name, TB.Name, TB.ID
FROM TA
FULL JOIN TB
ON TA.Name = TB.Name;
```

TA.ID	TA.Name	TB.Name	TB.ID
1	Pirate	Pirate	2
2	Monkey	NULL	NULL
3	Ninja	Ninja	4
4	Spaghetti	NULL	NULL
NULL	NULL	Darth Vader	3
NULL	NULL	Rutabaga	1

All values of two Tables(TA, TB) are printed.

Outer Join - Practice

Find the names of customers who don't have cards.

Customers

CustID	**Name**
1	John Smith
2	Jane Doe
3	Michael Scott
4	Pam Beesly
5	Dwight Schrute

Cards

CardID	**CustID**
1001	1
1002	2
1003	4

Outer Join - Practice

Find the names of customers who don't have cards.

Customers

CustID	Name
1	John Smith
2	Jane Doe
3	Michael Scott
4	Pam Beesly
5	Dwight Schrute

Cards

CardID	CustID
1001	1
1002	2
1003	4

```
SELECT C.Name
FROM Customers C
LEFT JOIN Cards CD
ON C.CustID = CD.CustID
WHERE CD.CardID IS NULL;
```

Name
Michael Scott
Dwight Schrute

Grouping – Practice

Write a query to calculate the cumulative quantity of inventory for each product and date based on the following Inventory table.

Inventory

InDate	ProductID	Qty
2024-10-01	101	50
2024-10-02	101	30
2024-10-03	101	20
2024-10-01	102	60
2024-10-02	102	40
2024-10-03	102	50

Result

ProductID	InDate	Qty	CQty
101	2024-10-01	50	50
101	2024-10-02	30	80
101	2024-10-03	20	100
102	2024-10-01	60	60
102	2024-10-02	40	100
102	2024-10-03	50	150

Grouping – Practice

Write a query to calculate the cumulative quantity of inventory for each product and date based on the following Inventory table.

Inventory

InDate	ProductID	Qty
2024-10-01	101	50
2024-10-02	101	30
2024-10-03	101	20
2024-10-01	102	60
2024-10-02	102	40
2024-10-03	102	50

Result

ProductID	InDate	Qty	CQty
101	2024-10-01	50	50
101	2024-10-02	30	80
101	2024-10-03	20	100
102	2024-10-01	60	60
102	2024-10-02	40	100
102	2024-10-03	50	150

```
SELECT A.ProductID, A.InDate, A.Qty, SUM(B.Qty) AS CQty
FROM Inventory A, Inventory B
WHERE A.ProductID = B.ProductID
AND A.InDate >= B.InDate
GROUP BY A.ProductID, A.InDate, A.Qty
```